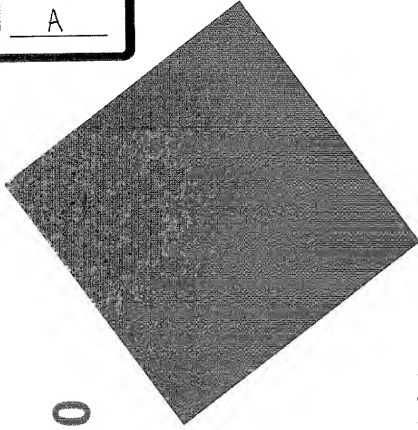
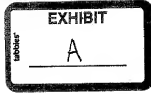




Generating Dynamic Web Pages with Net.Data for OS/390

Mel Zimowski
Net.Data for OS/390
IBM Software Solutions
zimowski@us.ibm.com

<http://www.software.ibm.com/data/net.data>



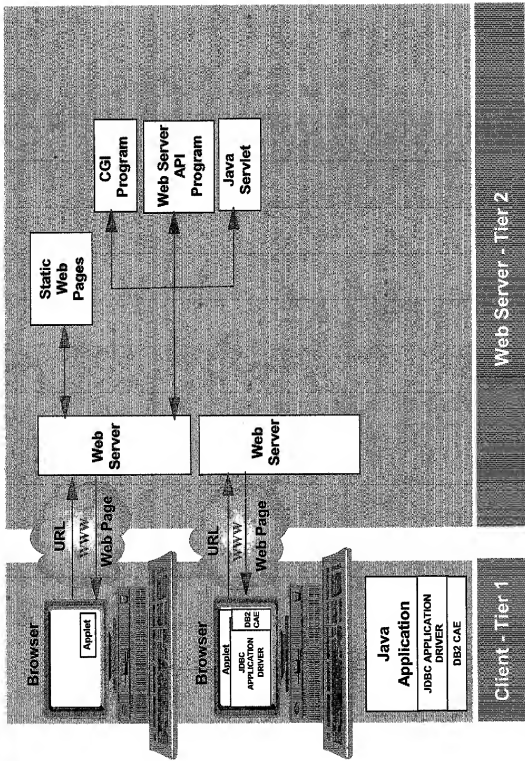
IBM Software

Agenda

- Types of DB2 web applications
- What is Net.Data for OS/390?
- How does Net.Data for OS/390 work?
- Installation
- Net.Data for OS/390 V2 enhancements
- Thoughts about future enhancements
- Product availability
- V1 and V2 software pre-requisites
- Sample Net.Data query utility macro

IBM Software

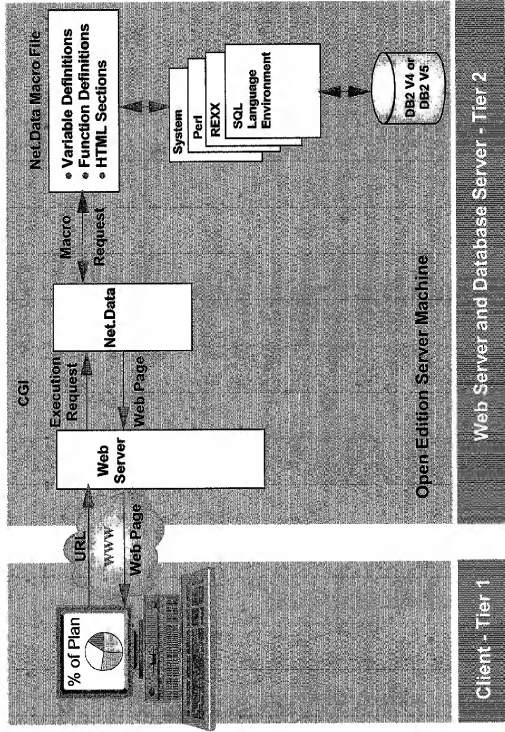
Types of DB2 Web Applications



<http://www.redbooks.ibm.com/> Accessing DB2 for OS/390 Data from the World Wide Web (SG24-5273-00)

IBM Software

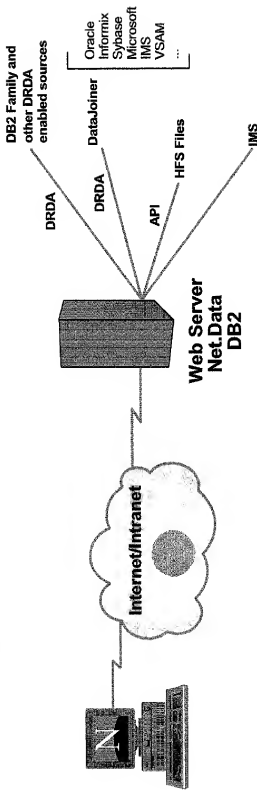
Net.Data for OS/390 Version 1



IBM Software

■ Access to the data in your enterprise

- Data managed by the local DB2 subsystem
- Other DB2 Family data through DRDA
- Other DRDA enabled data sources
- Heterogeneous data through DB2 DataJoiner
- HFS files
- IMS data

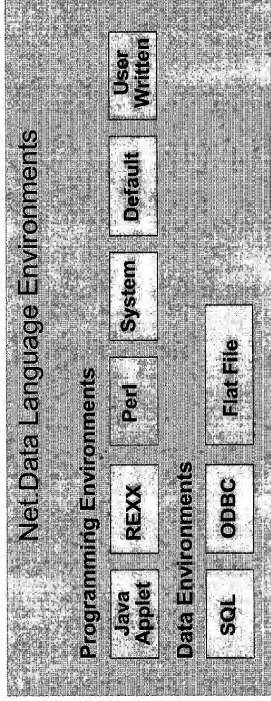


IBM Software

■ **Net.Data-provided programming environments**

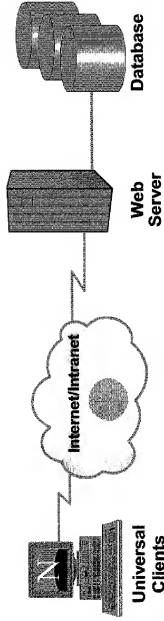
- Java applets
- REXX programs
- Perl scripts
- C/C++ applications
- Net.Data built-in functions

■ **API for user-provided language environments**



IBM Software

- **Full-featured macro capability for application flexibility**
- **Variable substitution for manipulation of input and output data**
- **If and While control statements**
- **Built-in libraries for developer productivity**
 - General (e.g. date, time, get environment variable)
 - Math (e.g. add, subtract, multiply, divide, power)
 - String (e.g. assign, concatenation, insert, substring, length)
 - Word (e.g. delete word, word count)
 - Table (e.g. generate HTML checkbox or radio input tags from a table variable)
- **Java applets and JavaScript programs for client-side processing**



IBM Software

Net.Data Macro File Building Blocks

Comment Section

Define Section

Function Definition Section

- Exec Section
- Report Section
 - Row Section
- Message Section
- Variable References

HTML Section

- Any HTML tags
- Function call
- Conditional Section
- Include Section
- Variable References

Message Section

IBM Software

Simple Net.Data Macro for Obtaining Stock Quotes

```
%FUNCTION(DTW_SQL stock_price(IN sym) {
  SELECT curprice FROM mrzstocklist WHERE symbol = $(sym) }
```

```
%REPORT {
%ROW {
  $(V1)
%}
%}
```

```
%}
```

```
%HTML(Symbol_Input) {
```

```
<Body>
```

```
<p>Today's date is @DTW_rDATE("U")</p>
```

```
<form method="post" action="New_Symbol">
```

```
<p>Enter stock symbol:
```

```
<input type="text" name="nsym"></p>
```

```
<p><input type="submit" value="Get Quote"></p>
```

```
</form>
```

```
</Body>
```

```
%}
```

```
%HTML(New_Symbol) {
```

```
<Body>
```

```
<p>The stock price for $(sym) is @stock_price(nsym)</p>
```

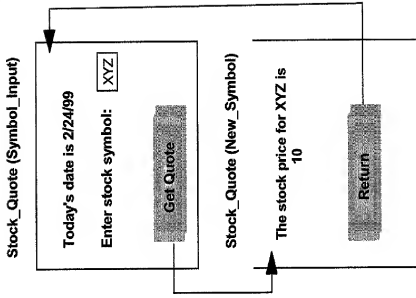
```
<form method="post" action="Symbol_Input">
```

```
<p><input type="submit" value="Return"></p>
```

```
</form>
```

```
</Body>
```

```
%}
```



IBM Software

How Does Net.Data for OS/390 Work?

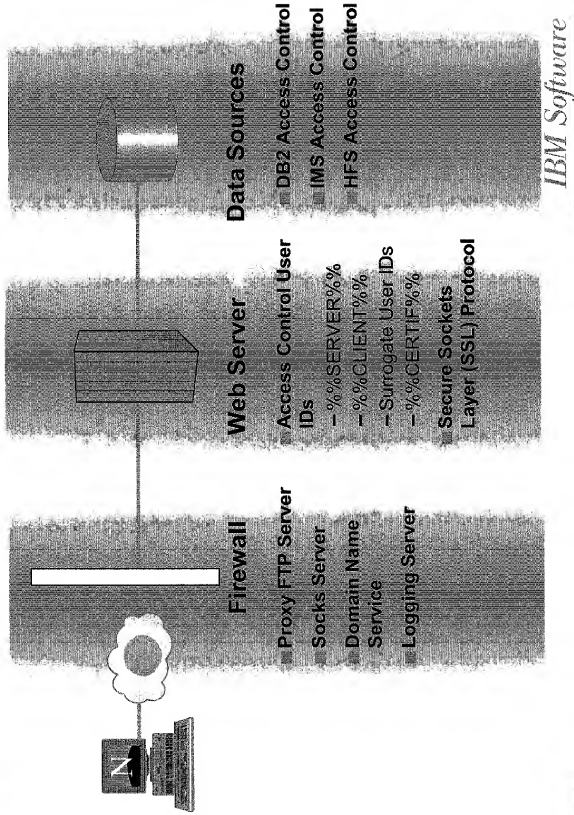
■ Security

■ Default initialization file

■ Frontend processing

■ Language Environment processing

Net.Data for OS/390 Security



Net.Data for OS/390 Initialization File

```
MACRO_PATH /usr/lpp/netdata/macros;  
EXEC_PATH /usr/lpp/netdata/testcmd;  
INCLUDE_PATH /usr/lpp/netdata/include;  
FFI_PATH /usr/lpp/netdata/file-data;  
DBSSID DBNC  
DB2PLAN DTWGA21  
ENVIRONMENT (DTW_SQL) dtwsql ()  
ENVIRONMENT (DTW_DEFAULT) defcdll (OUT RETURN_CODE)  
ENVIRONMENT (DTW_SYSTEM) sysdll (OUT RETURN_CODE)  
ENVIRONMENT (DTW_PERL) perldll (OUT RETURN_CODE)  
ENVIRONMENT (DTW_REXX) rexxdll (OUT RETURN_CODE)  
ENVIRONMENT (DTW_FILE) filedll (OUT RETURN_CODE)  
ENVIRONMENT (DTW_APPLET) appldll (OUT RETURN_CODE)  
ENVIRONMENT (DTW_ODBC) odbcdll (OUT RETURN_CODE)
```

- **Get environment variables (PATH_INFO, PATH_TRANSLATED, QUERY_STRING, REQUEST_METHOD, SCRIPT_NAME, SERVER_PORT)**
- **Determine document root directory for web server**
- **Determine path to initialization file**
- **Parse initialization file to obtain:**
 - Path information
 - Configuration variable values (e.g. Default DB2 SSID and Default DB2 plan name)
 - Language Environment configuration information
- **Parse URL**
- **Parse user input**
- **Parse and execute macro**

■ **Net.Data**

- Parse function call
- Associate function call arguments with function definition parameters
- Perform variable substitution on executable statements
- Build parameter list for language environment
- Invoke language environment API
- Update Net.Data variable values from output parameters
- Process %REPORT and %MESSAGE blocks
- Return function call return value

■ **Language Environment**

- Provide parameters to language interpreter
- Invoke language interpreter
- Retrieve output parameters from language interpreter

IBM Software

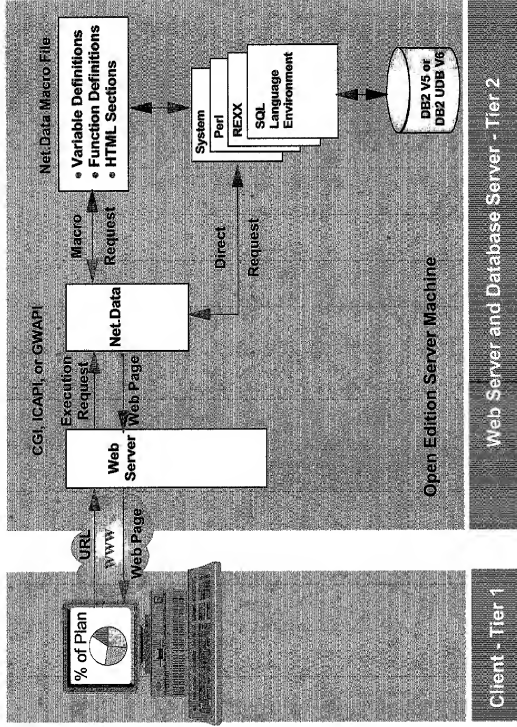
■ **SMP/E Installable**

- Unload the Sample JCL from the Product Tape
- Update and Execute the Desired Allocation Jobs
- Create the Hierarchical File System Structure
- Update and Execute the SMP/E RECEIVE Job
- Update and Execute the SMP/E APPLY CHECK and APPLY Job
- Update and Execute the SMP/E ACCEPT CHECK and ACCEPT Job
- Install the Net.Data Executables and DLLs
- Install and Customize the Net.Data Initialization File
- Set up the Language Environments
- Enable the Message Catalog

■ **Installation verification process**

- Create the Sample Database(s)
- Customize the Sample Macros
- Install the Sample Stored Procedures
- Test the Installation

Net.Data for OS/390 Version 2



IBM Software

Net.Data for OS/390 V2 Enhancements

Performance

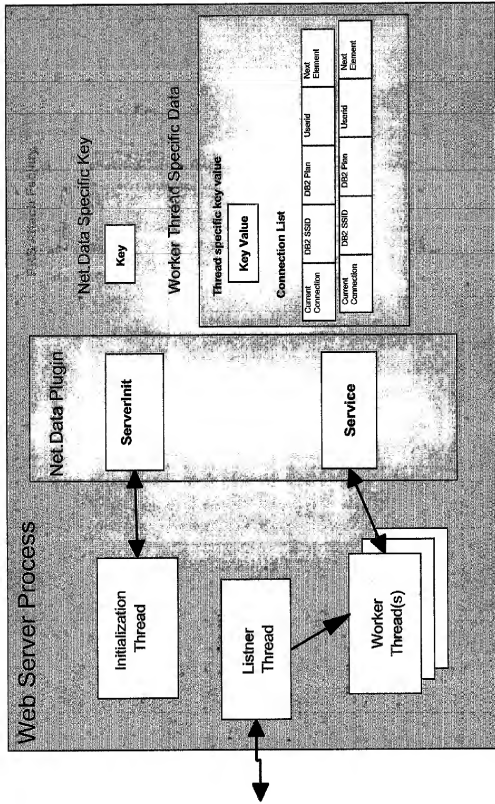
- ICAPI and GWAPI
- Management of DB2 Connections (SQL LE)
- Caching of initialization file
- Caching of Net.Data DLLs
- Invocation of Net.Data using direct requests
- DB2 message text lookup enhancement
- Whitespace enhancement

Scalability

- Integration with workload management

IBM Software

Management of DB2 Connections



IBM Software

■ **A direct request invokes Net.Data by specifying:**

- The name of a language environment
- An SQL statement, the name of a stored procedure, or the name of a REXX, Perl, C, or C++ function, along with any parameter values that are required for the invocation of the stored procedure or function
- Form data that is required for invocation of the SQL statement or function

■ **Stored procedure example:**

[http://server/netdata-cgi/db2www/?LANGENV=DTW_SQL&
FUNC=MY_STORED_PROC\(IN+CHAR\(30\)\)+Salaries\)](http://server/netdata-cgi/db2www/?LANGENV=DTW_SQL&FUNC=MY_STORED_PROC(IN+CHAR(30))+Salaries))

- Get environment variables (PATH_INFO, PATH_TRANSLATED, QUERY_STRING, REQUEST_METHOD, SCRIPT_NAME, SERVER_PORT)
- Determine document root directory for web server
- Determine path to initialization file
- Parse initialization file to obtain:
 - Path information
 - Default DB2 SSID
 - Default DB2 plan name
 - Language Environment configuration information
- Parse URL including direct request

■ **Net.Data**

- Build parameter list for language environment
- Invoke language environment API
- Perform required output variable and report processing

■ **Language Environment**

- Provide parameters to language interpreter
- Invoke language interpreter
- Retrieve output parameters from language interpreter

■ **DB2MSGSG configuration variable**

Syntax: DB2MSGSG [=] message_level

Where message_level indicates the level of DB2-provided messages that Net.Data displays:

- NONE
Specifies that Net.Data displays no message text.
- ERRORONLY
Specifies that Net.Data displays message text only for negative SQLCODE values.
- ALL
Specifies that Net.Data displays message text for all SQLCODE values. This is the default.

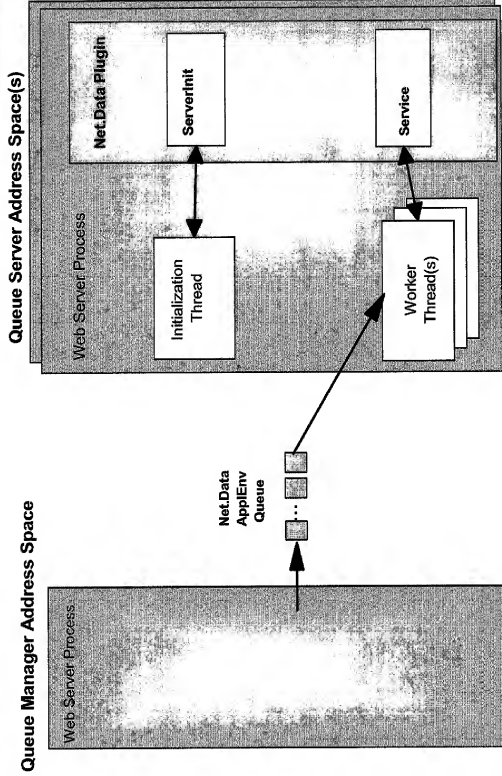
■ **Reduces or eliminates the cost of obtaining message text associated with SQLCODE values**

IBM Software

■ DTW_REMOVE_WS configuration variable

Syntax: DTW_REMOVE_WS [=] YES|NO

- Reduces the size of a dynamically generated Web page by compressing any sequence of two or more white space characters (any combination of tabulators, blanks, and new-line characters) into one new-line character. The default is NO.



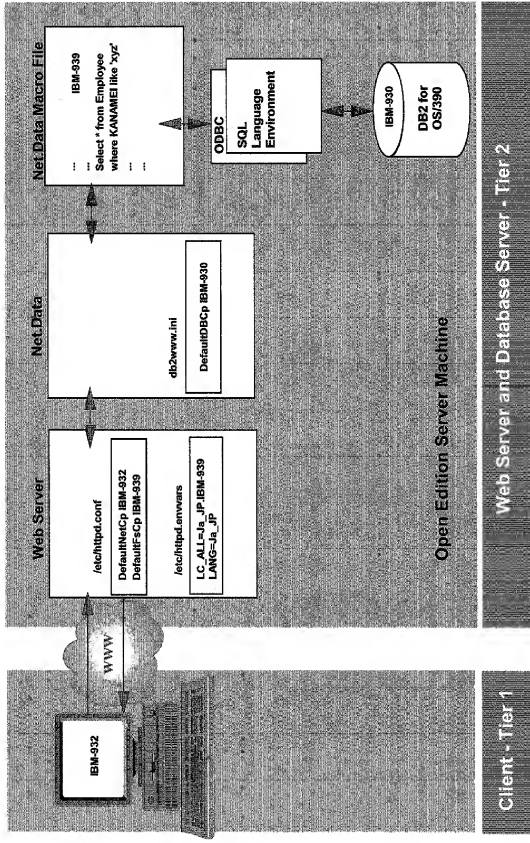
Macro Language Enhancements

- Nested IF and WHILE control sections
- Support for the return of a single result set from a stored procedure
- DBCS enabled string and word functions
- Maximum include file and function sizes increased to 256KB
- Comments anywhere

Miscellaneous

- Ability to construct web page from HTML text and database data in different code pages

Building Web Pages with Input from Different Code Pages



IBM Software

Performance

- Caching of dynamic web pages
- Caching of macro files
- Management of DB2 Connections (ODBC LE)
- Built-in functions performance enhancement
- Parser efficiency
- Performance monitor

Scalability

- Improve scalability of table variables for large result sets
- WLM Enhancements for waiting for system resources

- ASCII encoded web pages are cached in DB2 for reuse by Net.Data
- ICAP-only enhancement
- Only complete web pages are cached
- Caching directives specify the web pages to be cached and the number of seconds the cached page is valid
- Key of cached page consists of URL including query string plus form data (if present)
- Cached web pages are classified as PUBLIC or PRIVATE

- **Macros and %INCLUDE files are cached in memory for reuse by the parser**
- **ICAPI-only enhancement**
- **One cache per web server address space**
- **Caching directives specify the macros to be cached**
 - **DTW_CACHE_MACRO** - specifies macros that are to be cached
 - **DTW_DO_NOT_CACHE_MACRO** - specifies macros that are not to be cached
- **Security context of thread used to determine authorization of requester to execute cached macro**

■ Macro Language Enhancements

- Ability to receive multiple result sets from a stored procedure
- Built-in functions for table processing
- Built-in functions that allow macros to get and set Netscape cookies
- Functions consisting of macro language statements
- EuroCurrency
- Support for START_ROW_NUM
- Support for DTW_SET_TOTAL_ROWS and TOTAL_ROWS
- Support for DTW_EXIT
- Built-in function that allows a macro to send e-mail
- XML

IBM Software

```
%FUNCTION(DTW_SQL) stored_proc() {  
CALL my_proc(result_table_1,result_table_2)  
%REPORT(result_table_2){  
.  
.  
.  
%}  
%REPORT(result_table_1){  
.  
:  
%}  
%}  
%HTML(OUTPUT) {  
.  
.  
.  
@stored_proc()  
.  
.  
.  
%}
```

- **DTW_TB_APPENDROW** - adds rows to the end of a table
- **DTW_TB_COLS** - returns the current number of columns in a table
- **DTW_TB_DELETECOL** - deletes columns from a table
- **DTW_TB_DELETEROW** - deletes rows from a table
- **DTW_TB_GETN** - returns the name of a column associated with a column number
- **DTW_TB_GETV** - returns a column value from a row
- **DTW_TB_INSERTCOL** - inserts new columns in a table
- **DTW_TB_INSERTROW** - inserts new rows into a table
- **DTW_TB_MAXROWS** - returns the maximum number of rows allowed in a table
- **DTW_TB_QUERYCOLNOJ** - returns the column number associated with the name of a column
- **DTW_TB_ROWS** - returns the current number of rows in a table
- **DTW_TB_SETCOLS** - sets the number of columns in a table
- **DTW_TB_SETN** - assigns a name to a column
- **DTW_TB_SETV** - assigns a column value to a row

■ **DTW_SETCOOKIE - sets the value of a Netscape cookie**

Syntax: @DTW_SETCOOKIE (IN cookie_name, IN cookie_value, IN advanced_options)

where advanced_options specifies values for the following cookie attributes:

- expires - lifetime of the cookie
- domain - domains for which cookie is valid
- path - URLs for which cookie is valid
- secure - transmit only to HTTPS servers using SSL protocol

■ **DTW_GETCOOKIE - returns the value of a Netscape cookie**

Syntax: @DTW_GETCOOKIE (IN cookie_name, OUT cookie_value)

■ **Netscape HTTP Cookies specification:**

http://home.netscape.com/newsref/std/cookie_spec.html

Functions Consisting of Macro Language Statements

- **Permits a macro to invoke a subroutine that consists entirely of Net.Data macro language statements**

```
%AMACRO_FUNCTION setMessage (IN rc, OUT message) {  
  %IF (rc == "0")  
    @dtw_assign(message, "Function call was successful")  
  %ELIF (rc == "-1")  
    @dtw_assign(message, "Function failed, out of memory")  
  %ELIF (rc == "-2")  
    @dtw_assign(message, "Function failed, invalid parameter")  
  %ENDIF  
  %}
```

IBM Software

Built-in Function to Send E-mail

- **Permits a macro to dynamically build and transmit e-mail messages**

Syntax: @DTW_SENDMAIL (IN sender, IN recipient, IN message, IN subject,
IN carbon_copy, IN blind_carbon_copy, IN reply_to,
IN organization)

```
%Function(DTW_SQL mailing_list (IN message) {  
  SELECT EMAIL_ADDRESS FROM CUSTOMERS WHERE STATE='CA'  
  %REPORT {  
    Sending out product information to all customers who live in California...<P>  
    %ROW {  
      @DTW_SENDMAIL ("John Doe Corp. <John.Doe@doe.com>", V1, message,  
        "New Product Release")  
      E-mail sent out to customer $(V1).<BR>  
      %}  
    %}  
  %}
```

IBM Software

Thoughts about Future Net.Data for OS/390 Enhancements

Language Environment Enhancements

- Support for DB2 large objects
- IMS transactions Language Environment

Miscellaneous

- Ability to execute Net.Data as a Servlet
- Net.Data Message Log File
- Net.Data Trace
- Eliminate DTWIODLL
- Application-development tool
- Serviceability aids

IBM Software

Net.Data for OS/390 Product Availability

Net.Data for OS/390 V1

- GA for English - June 1997
- GA for Japanese - October 1997

Net.Data for OS/390 V2

- GA for English, Japanese, and Korean - June 1998

V1 Software Prerequisites (Minimum Release Levels)

■ **Operating System**

- MVS/ESA Version 5.2.2 (5655-068 or 5655-069) or
- OS/390 Version 1 Release 1 (5645-001)

■ **HTTP Web Server**

- Any HTTP-compliant Web Server, or
- IBM Internet Connection Server Version 1 Release 1 for OS/390 (5655-156), or
- IBM Internet Connection Secure Server Version 2 Release 1 for OS/390 (5697-B14)

■ **Database Management System**

- DB2 Server for OS/390 Version 5 (5655-DB2) and the PTF for APAR PQ09901, or
- DB2 for MVS/ESA Version 4 (5695-DB2)

Note: DB2 Server for OS/390 Version 5 is a prerequisite for use of ODBC

■ **Language Environment**

- Language Environment 370 Version 1.5 (5688-198) and the PTFs for APARs PN80033, PN79307, PN80015, and PN76475

IBM Software

V2 Software Prerequisites (Minimum Release Levels)

■ **Operating System**

- OS/390 Version 1 Release 3 Base Services (5645-001)
- OS/390 Version 1 Release 3 Application Enablement optional feature for DFSORT

■ **HTTP Web Server**

- IBM Internet Connection Secure Server for OS/390 Version 2 Release 2 (5697-B14) including the PTFs for APARs PQ15294 and PQ05981, or
- Domino Go Webserver 4.6.1 for OS/390 (5697-C58) and the PTF for APAR PQ15294, or
- Domino Go Webserver 5.0 for OS/390 (5697-D43)

■ **Database Management System**

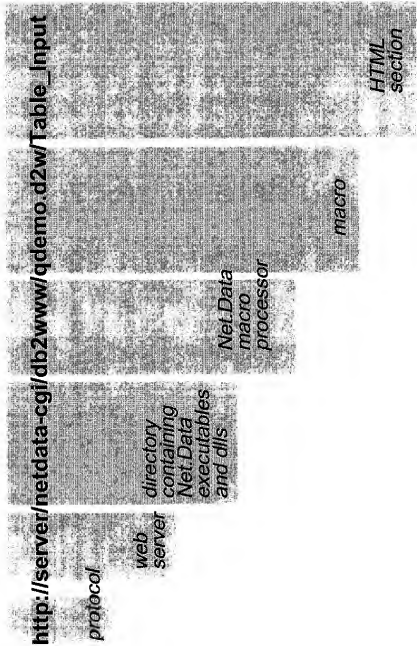
- DB2 Server for OS/390 Version 5 (5655-DB2) and the PTFs for APARs PQ06950 and PQ09901, or
- DB2 Universal Database for OS/390 V6 (5645-DB2)

■ **Language Environment**

- Language Environment 370 Version 1.5 (5688-198) and the PTFs for APARs PN80033, PN79307, PN80015, and PN76475

IBM Software

URL to Execute Macro qdemo.d2w



IBM Software

Define section for Macro qdemo.d2w

```
%{ ***** Define Section ***** %}  
%DEFINE {  
    DTW_DEFAULT_REPORT="NO"  
    coltab=%TABLE  
    alltab=%TABLE  
    individualtab=%TABLE  
    %LIST " " colnames  
%}
```

Function Definition Section for Macro qdemo.d2w

```
%{ ***** Function Definition Section ***** %}

%FUNCTION(DTW_SQL) selectall(OUT outtab) {
  SELECT * FROM $(input_tabnam)

  %MESSAGE {
    -104 : "<STRONG>Table name not specified</STRONG>" ; continue
    -204 : "<STRONG>Table not found</STRONG>" ; continue
  }

  %}

%FUNCTION(DTW_SQL) selectcolumnnames(OUT outtab) {
  SELECT NAME, COLTYPE FROM SYSIBM.SYSCOLUMNS
  WHERE TBcreator = $(creator) and TBNAME = $(tabnam)

  %MESSAGE {
    100 : "<STRONG>Table not found</STRONG>" ; exit
    -104 : "<STRONG>Table name not specified</STRONG>" ; continue
    -204 : "<STRONG>Table not found</STRONG>" ; continue
  }

  %}

%FUNCTION(DTW_SQL) selectindividual(OUT outtab) {
  SELECT $(colnames) FROM $(input_tabnam)

  %MESSAGE {
    -104 : "<STRONG>Individual columns not selected</STRONG>" ; continue
    -204 : "<STRONG>Table not found</STRONG>" ; continue
  }

  %}

%FUNCTION(DTW_PERL) todays_date() RETURNS(result) {
  $date = `date` ;
  chop $date ;
  open(DTW, "> $ENV(DTWPIPE)") || die "Could not open: $!" ;
  print DTW "result = \"$date\\n" ;
  %}

%}
```

IBM Software

HTML Section for Macro qdemo.d2w: Table__Input

```
%{ ***** HTML Section: Table_Input ***** %}  
  
%HTML(Table_Input) {  
<Title>Net.Data Query Utility</Title>  
<Body>  
<hr>  
<h1 align=center>Table Selection</h1>  
<hr>  
<br>  
<form method="post" action="Process_Column_Set_Input">  
<p>Enter Table Name: <input type="text" name="input_tabnam"></p>  
<p>Select Column Set:  
<SELECT NAME="columnset">  
<OPTION SELECTED>All Columns  
<OPTION>Individual Columns  
</SELECT></p>  
<p><input type="submit"></p>  
<p>SHOWSQL: <input type="radio" NAME="SHOWSQL" VALUE="YES"> Yes  
          <input type="radio" NAME="SHOWSQL" VALUE="" CHECKED> No  
</p>  
</form>  
</Body>  
%}
```

IBM Software

HTML Section for qdemo.d2w: Process Column Set Input

```
%{ ***** HTML Section: Process_Column_Set_Input ***** %}  
  
%HTML (Process_Column_Set_Input) {  
<Title>Net.Data Query Utility</Title>  
<Body>  
<br>  
%IF $(columnset) == "Individual Columns"  
    @DTW_POS(" ",input_tabnam,period_position)  
%IF $(period_position) == "0"  
    @DTW_GETENV("_BPX_USERID",creator)  
%ELSE  
    @DTW_SUBTRACT(period_position,"1",creator_end_position)  
    @DTW_SUBSTR(input_tabnam,"1",creator_end_position,creator)  
%ENDIF  
    @DTW_UPPERCASE(creator,creator)  
    @DTW_CONCAT(" ",creator,creator)  
    @DTW_CONCAT(creator," ",creator)  
    @DTW_ADD(period_position,"1",tabnam_start_position)  
    @DTW_SUBSTR(input_tabnam,tabnam_start_position,tabnam)  
    @DTW_UPPERCASE(tabnam,tabnam)  
    @DTW_CONCAT(" ",tabnam,tabnam)  
    @DTW_CONCAT(tabnam," ",tabnam)  
    @selectcolumnnames(coltab)
```

IBM Software

HTML Section for qdemo.d2w: Process Column_Set_Input

```
<hr>
<h1 align=center>Individual Column Selection</h1>
<hr>
<br>
<form method="post" action="Individual_Column_Output">
<input type="HIDDEN" NAME="input_tabnam" VALUE="$ (input_tabnam)">
<p>Select Individual Columns:
<br>
<p><@DTW_TB_INPUT_CHECKBOX(coltab,"1","colNams","1")></p>
<p><input type="submit"></p>
<p>SHOWSQL: <input type="radio" NAME="SHOWSQL" VALUE="YES"> Yes
<input type="radio" NAME="SHOWSQL" VALUE="" CHECKED> No
</p>
</form>
%ELSE
@DTW_UPPERCASE(input_tabnam,input_tabnam)
<hr>
<h1 align=center>Query Results</h1>
<br>
<p>@today_s_date()
<br>
<p>@selectall(alitab)
@DTW_TB_TABLE(alitab,"BORDER")
<form method="post" action="Table_Input">
<p><input type="submit" value="Query Another Table"></p>
</form>
%ENDIF
</Body>
%}
```

IBM Software

HTML Section for qdemo.d2w: Individual_Column_Output

```
%{ ***** HTML Section: Individual_Column_Output ***** %}

%HTML(Individual_Column_Output) {
<Title>Net.Data Query Utility</Title>
<Body>
<hr>
<h1 align=center>Query Results</h1>
<hr>
<br>
@DTW_UPPERCASE(input_tabnam, input_tabnam)
<p>@today's_date()
<br>
<p>@selectindividual(individualtab)
@DTW_TB_TABLE(individualtab, "BORDER")
<form method="post" action="Table_Input">
<p><input type="submit" value="Query Another Table"></p>
</form>
</Body>
%}
```

IBM Software